Remarks

In the non-final Office Action dated November 24, 2008, claims 1-4, 9-13, 18-21, and 25 are pending, and claims 1-4, 9-13, 18-21, and 25 stand rejected. The Applicants have not amended the claims in this Response. The Applicants respectfully traverse the rejections herein.

35 USC § 101 Rejection

The Examiner rejected claims 1-4 and 9 under 35 USC § 101, suggesting that the claims are directed to non-statutory subject matter. The Applicants respectfully disagree.

Claim 1 recites a process of transforming datastreams from a first format to a second format. The process recited in claim 1 provides specific steps to perform this activity, including parsing a datastream into work units in a first format, and converting the first format work unit into a second format. In in re Bilski (CAFC October 30, 2008, no citation available) the court noted that "a claimed process is patent-eligible if it transforms an article into a different state or thing (Page 24)", reaffirming the machine-or-transformation test provided by the Supreme Court (Diamond v. Diehr, 450 U.S. 175; Benson U.S. 409). The Applicants submit that one skilled in the art will readily recognize that transforming one kind of datastream into a different kind of datastream is clearly a different state or thing. For example, one skilled in the art would understand a process of converting a MO:DCA datastream into IPDS is a transformation from one thing to another thing. One skilled in the art would also understand other statutory processes, such as converting un-encrypted data to encrypted data, converting PDF documents to word documents, or converting TIFF files to GIF files transforms underlying datastreams (articles) from one thing to another thing. Thus, the Applicants submit that claim 1 recites a patent eligible process and respectfully request withdrawal of the § 101 rejection.

35 USC § 103(a) Rejection

The Examiner rejected claims 1-4, 9-13, 18-21, and 25 under 35 USC § 103(a) as being obvious in view of U.S. Patent No. 6,690,478 (McIntyre) in various combinations with U.S. Patent Application Publication No. 2004/0243934 (Wood) and U.S. Patent No. 6,504,621 (Salgado). The Applicants submit that the claims are non-obvious over the cited art.

Claim 1, paraphrased herein, recites a method for transforming a datastream. According to the method, a datastream from a single job is parsed into a plurality of work units in a first format, where each work unit may be processed independent of all other work units. The work units may be either a data work unit or a control work unit. Each control work unit may be an intermediate control work unit or a schedule control work unit. Each data work unit is queued on a queue accessible to a plurality of compute nodes. Scheduled control work units are queued at a tail of the queue to be processed by a compute node after all other work units presently in the queue. An immediate control work unit is queued at a head of the queue to be processed by a compute node before all other work units in the queue. An interrupt control work unit is forwarded to a compute node immediately regardless of any work units in the queue. Each of the plurality of work units are processed by at least one compute node to convert each data work unit into a second format, where the processing of each work unit is independent of processing of the other work units and where multiple work units are processed in parallel by multiple compute nodes.

First, the Applicants submit that none of the cited art teaches or reasonably suggests "wherein each work unit may be processed independent of all other works units" as recited in claim 1. In Wood, paragraphs 4-6 discuss known techniques for parsing a PDL datastream into multiple segments. As discussed in Wood, a PDL datastream includes data commands and control commands. Data commands describe the data to be reproduced at an output device, such as text data, graphics data, and image data. Control commands describe how the data is reproduced on the output devices. Examples of control commands include font information, page sections, forms, and overlays. Wood further discloses that each segment must "know" the appropriate "translation state" for their respective segment. A "translation state" is composed of all of the previous control commands from the PDL datastream. For example, if the PDL datastream was parsed into two segments, then one segment would be required to "know" all the control commands associated with the other segment. Wood discusses solving this requirement by including all the previous commands in each segment during the parsing process. For example, a first segment parsed from a PDL datastream would include data and control commands for data within the first segment. A second segment parsed from the PDL datastream would include all the control commands from the first segment and the control commands for the data within the second segment.

The Examiner suggests that because each segment includes the appropriate "translation state", each segment may be processed independently of all other segments. The Applicants respectfully disagree. Referring to our two segment example previously provided, at least one of the segments is required to include all the control commands in the other segment. Because of this, the segments are dependent upon each other. Thus, the segments cannot be "processed independently of all other" segments. For example, if the PDL datastream was parsed into two segments and neither segment contained the required "translation state" information, then the system disclosed in Wood would not be operable to process the segments to generate intermediate data. It would not be operable to do so because at least one of the segments would not contain the required "translation state" information which is only available from the other segment of data.

The Examiner additionally suggests that a further indication of segment independence is shown by the various portions of the intermediate data being combined into a single intermediate datastream. The Examiner suggests that, because the various portions may be generated at different rates, the processing must be independent. The Examiner suggests this because generating the intermediate data in may not be performed in chronological order. Because the intermediate data is not generated in chronological order, the system would have to rearrange the intermediate data to obtain an intermediate datastream in chronological order. The Applicants respectfully disagree that this disclosure teaches that the processing is independent.

Wood discloses that intermediate datastream portions may be generated out of order. That is, some datastream portions may be generated before other datastream portions. Wood additionally discloses that the datastream portions are generated in parallel. The Applicants submit that out of order generation does not require independent processing. For example, some processors may simple be generating data slow than other processors. In some cases, some segments may be more computationally difficult that other segments. In other cases, the datastream may be processed in reverse order (e.g., the last segment is processed first). Because the last segment requires the "translation state" information from all the previous segments, processing the datastream in reverse order may be more efficient. In such a case, intermediate data would be generated in reverse chronological order, which would require the reordering suggested in Wood.

The Applicants further submit that Wood does not teach "wherein each work unit may be

processed independent of all other works units" as recited in claim 1. Wood discloses methods and apparatus for processing a PDL datastream. In Wood, the PDL datastream is parsed into a plurality of segments. The segments are assigned for processing on at least one PDL processor (Abstract). Wood additionally discloses that two data files are created for each segment created (FIG. 4; Paragraph 29). The first data file is a global data file. Global data files include all the PDL commands from the associated segment that may affect subsequent segments. Global data files contain information similar to the "translation state" information as described above, except in Wood the "translation state" information is contained in a separate file instead of being included in each data segment. Similar to "translation states", global data files include font settings, and forms for the segments. The next data file created for each segment is a data file. Each data file includes all of the PDL commands necessary to interpret the data file associated with the segment. Wood further discloses in FIGS. 4-5 how the PDL processors process the different data files.

FIG. 4 illustrates how a scheduler 24 receives the global data file 20 and the segment data file 22 and distributes the data files to a plurality of PDL processors 38_1 , 38_2, 38_3...38_m. In paragraph 34, Wood discloses:

> when assigning data files 20 and 22 to PDL processors 38_1 , 38_2 , 38_3 ...38_m, scheduling process 24 accounts for the order-dependence of the data files. That is, before a PDL processor may process a specific segment data file (e.g., segment file 22_k), the processor must first process either the global data files 20 or segment data files 22 for all previous segments (i.e., global data files 20_1 , 20_2 , ... , 20_{K-1} or segment data files 22_1 , 22_2 ,. . . , 22_{K-1}).

The Examiner has suggested that Wood simply discloses that global files might affect subsequent segments during processing. The Applicants respectfully disagree. Wood illustrates PDL processing of data in FIGS. 5A, 5B, and 5C. In FIG. 5A, PDL processor 38_1 processes segment data files 1-5 in sequential order. Each segment data file is processed one at a time in a linear fashion. First segment data file 1 is processed. Next segment data file 2 is processed. Each of the remaining segment data files 3-5 are processed in a similar sequential manner. FIG. 5B illustrates using two PDL processors to process segment data files 1-5. PDL processor 1 processes segment data file 1, segment data file 4, and segment data file 5. After the PDL processor processes segment data file 1, global data file 2 and global data file 3 must be

processed before segment data file 4. In the FIG. 5B example, the PDL processor 1 does not processes segment data files 2 and 3. Because of this, the PDL processor must use global data files 2 and 3 associated with the missing segment data files. Similar to the prior art discussed by wood, global data files 2 and 3 form the "translation state" information required by the PDL processor to span from segment data file 1 to segment data file 4. Similar requirements are illustrated in FIGS 5B and 5C. The Applicants submit that both Wood and the prior art disclosed by Wood each have the same types of segment dependent processing requirements. In one case the "translation state" information is contained in each parsed segment. In the other case the "translation state" information is contained in the global data file. In neither case are the segments processed independent of other segments.

The Applicants further submit that neither McIntyre nor Salgado discloses this limitation.

For at least these reasons, the Applicants submit that none of the cited art by the Examiner teaches or reasonably suggests "parsing a datastream into a plurality of work units in a first format wherein each work unit may be processed independent of all other works units" as recited in claim 1.

Second, the Applicants submit that none of the cited art teaches or reasonably suggests "processing each of the plurality of work units by at least one compute node to convert each data work unit into a second format, wherein the processing of each work unit is independent of processing of the other work units and wherein multiple work units are processed in parallel by multiple compute nodes." As discussed above, none of the cited art discloses processing work units independent of processing other work units. Because none of the cited art discloses processing each segment independent of processing other segments, none of the cited art is operable to further convert each segment into a second format in claimed independent manner.

Third, the Applicants submit that the Examiner has improperly combined Salgado with McIntyre and Wood in rejecting the following limitations:

queuing a scheduled control work unit at a tail of the queue to be processed by a compute node after all other work units presently in the queue;

queuing an immediate control work unit at a head of the queue to be processed by a compute node before all other work units in the queue;

forwarding an interrupt control work unit to a compute node immediately regardless of

any work units in the queue;

Salgado discloses a queue management system for managing a queue of print jobs in a printing system. In Salgado, print jobs in a queue are held when resources required by the print job are unavailable (Summary, Background). Salgado provides examples whereby different print jobs in a queue may be processed, held, or reordered depending on the priority of the print jobs, and the available print resources (Example 1). In rejecting this limitation, the Examiner suggests that print jobs disclosed in Salgado are the "work units" recited in claim 1. The Applicants disagree. First, one skilled in the art of print job queue management would not confuse a print job with the plurality of "work units" recited in claim 1. Print Jobs disclosed in Salgado comprise complete files or sets of files for printing. "Work units" recited in claim 1 are the result of parsing a datastream comprising a single job. Furthermore, there is no teaching of parsing print jobs into a plurality of work units in Salgado. For example, Salgado does not teach that a datastream containing multiple print jobs is parsed into individual print jobs for processing. The Applicants therefore submit that the Examiner is using improper hindsight knowledge after reading the pending claims to piece together portions of Salgado with Wood and McIntyre to form the rejection. The Applicants will remind the Examiner that the pending claims cannot be used as a template to piece together isolated references and teachings so that the claimed invention may be rendered obvious. The claims must be viewed as they would have been perceived in the state of the art that existed at the time the invention was made.

For at least the reasons provided above, the Applicants submit that claim 1 is non-obvious in view of the cited art. Similar arguments apply for claims 10 and 19. Dependent claims 2-4, 9, 11-13, 17-18, 20-21, and 25 are non-obvious for at least depending on allowable base claims 10 and 19.

Conclusion

The Applicants submit that claims 1-4, 9-13, 18-21, and 25 are non-obvious in view of the cited art, and therefore respectfully request the Examiner allow claims 1-4, 9-13, 18-21, and 25.

Respectfully submitted,

Date: February 17, 2009

/Sean J. Varley/

Sean J. Varley, Reg. No. 62,397
Duft, Bornsen & Fishman, LLP
1526 Spruce Street, Suite 302
Boulder, CO 80302
(303) 786-7687
(303) 786-7691 (fax)
Customer Number: 50441